1. XGB Regressor
2. Practical with HyperParameter tuning (OPtuna)
3. Why XGB

{ XGB → opensource Library }

ML ⇒ 70s, 80s

Linear, Naive bayes

↓ Linear data

↓ Binary classification

for specific

→ 90's → RF, Svm, GB

Linear + non Linear

Generalize

1. overfitting
→ (Parameters)

2. (Scalability)

Speed

SE → OOB, Cache, distributed

XGB

2016

ML + SE ⇒ Libary ⇒ XGB

↓

GB

↑

better

XGB

Early Days → Kaggle competition → Open Source

2019, 2020

2011

{Gradient Boosting} Loss

2011 → 30 → 19 XGB

1. too many feature
2. optimization
3. Prod.
4. multiple Lang.

→ XGB Library

1. flexibility (Loss fn) (Differentiation)
   - reg
   - class
   - Ranking
   - Custom
2. Performance
3. Scalability
4. Robust
   - → missing
   - → outlier
   - → Regulization → overfitting

XGB Regressor

Cgpa	Package	BE	Res1
6.2	4.5	7.37	-3
9.0	11.0	7.37	3.5
7.5	6.0	7.37	-1.5
5.0	8.0	7.37	0.5

Base estimator + α DT

Stage1   Stage2

G.B. ⇒

XGB ⇒ DT → Diff ⇒ Vanila DT | Fundamental DT

Similarity Score, gain

Gini, entropy

Diff criteria

Cgpa	Package	Base Model (X) (Avg of target)	Residual
6.2	4.5	7.3	-2.8
9.0	11.0	7.3	3.7
7.5	6.0	7.3	-1.3
5.0	8.0	7.3	0.7

Cgpa | Res → DT → S/E

[-2.8, 3.7, -1.3, 0.7] ⇒ Similarity score

$$SS = \frac{(\text{Sum of Res})^2}{\text{No. of Res} + \lambda}$$

λ = hyperparameter (Reg. parameter)

$\lambda = 0$

DT {Gain ↑, S.L. ↑}

1. Sort the cgpa feature
2. Average of the consecutive value

5.0, 6.2 → 5.85
6.2, 7.5 → 7.1
7.5, 9.0 → 8.25

DT ⇒ Num ⇒ Independent

$$SS = \frac{(\text{Sum of Res})^2}{\text{no of Res} + 0} = \frac{SR}{\#Res}$$

$$= \frac{(-2.8 + 3.7 - 1.3 + 0.7)^2}{4} = \frac{(0.3)^2}{4} = 0.02$$

Splitting criteria

① 5.85

Root: CGPA < 5.85 ⇒ 0.02

child: 0.7 | -2.8, -1.3, 3.7

$$SS_1 = \frac{(0.7)^2}{1+0} = 0.49$$

$$SS_2 = \frac{(-2.8 - 1.3 + 3.7)^2}{3+0} = \frac{0.16}{3} = 0.05$$

Gain ⇒ $(SSL + SSR) - SS_{root}$
⇒ $0.49 + 0.05 - 0.02$
⇒ 0.52

② 7.1

Cgpa < 7.1

0.7, -2.8 | 3.7, -1.3

$$SSL = \frac{(0.7 - 2.8)^2}{2} = \frac{4.41}{2}$$

$SSR = 2.20$

$$SSR = \frac{(-1.3 + 3.7)^2}{2} = \frac{5.76}{2} = 2.88$$

$$\text{Gain} = SSL + SSR - S_{root} = (2.20 + 2.88) - 0.02 = 5.06$$

③ 8.25

Cgpa < 8.25

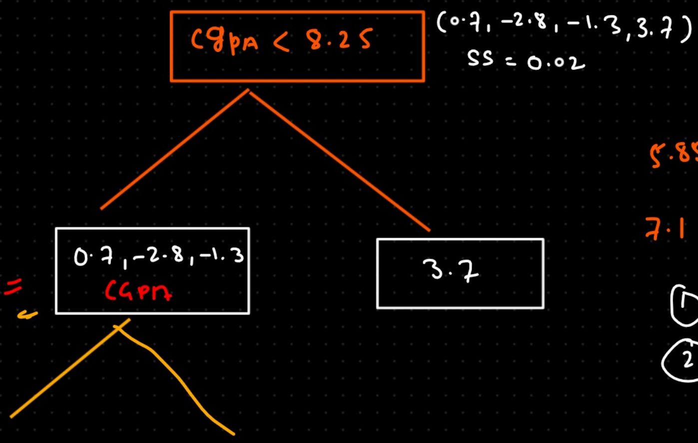
0.7, -2.8, -1.3 | 3.7

$SSL = 3.85$
$SSR = 13.69$
SS_{R} 0.02

Gain $= (3.85 + 13.69) - (0.02) = 17.52$

Highest Gain = 17.52 ⇒ Cgpa < 8.25

DT ⇒ AB ⇒ stump (max_depth 1)
GB = DP (8 - 32)
XGB ⇒ max_depth = 6

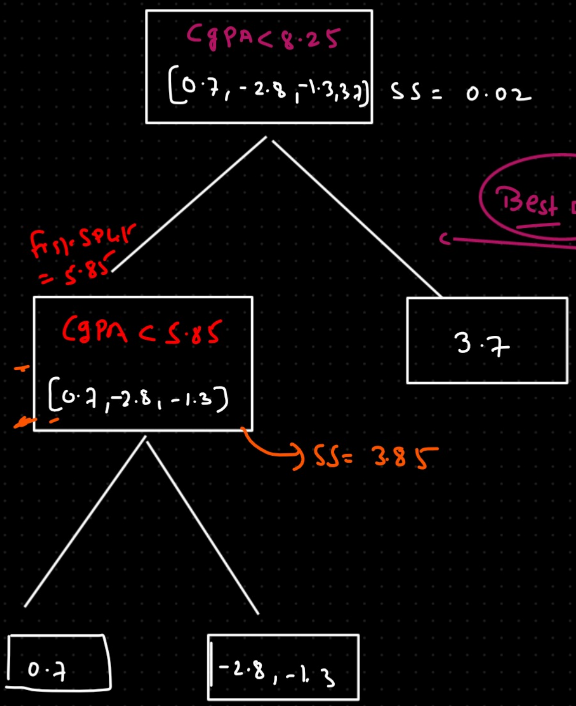


CGPA	Resi
5.0	0.7
6.7	-2.8
7.8	-1.3

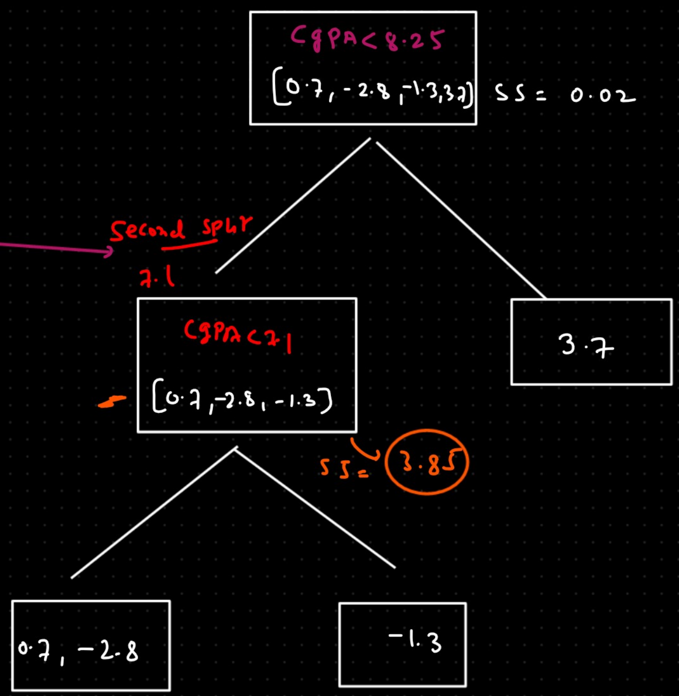
5.85 ←

7.1 ←

1. SS
2. Gain



Best one



SSL

$= \frac{(0.7)^2}{1}$

$= 0.49$

SSR

$= \frac{(-2.8 - 1.3)^2}{2}$

$= 8.40$

gain = [(SSL + SSR) - Root]

$= 0.49 + 8.40 - 3.85$

$= 5.04$

SSL

↓

$SSL = \frac{(0.7 - 2.8)^2}{2} = 2.20$

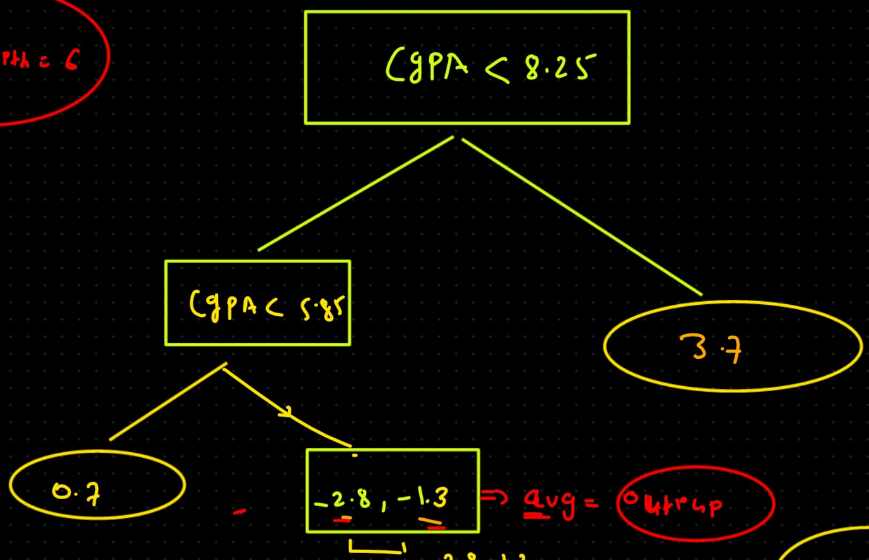
$SSR = \frac{(-1.3)^2}{1} = 1.69$

gain = ((SSL + SSR) - Root)

$= 2.20 + 1.64 - 3.85$

$= 0.04$

Max_depth = 6



⇒ avg = output

$\frac{-2.8 - 1.3}{2} = \frac{-4.1}{2} = -2.05$

Pakg_model

I/P	Target/Act	Pred1	Res1	Pred2	Res2
CGPA	Pakage	BaseModel1	Residual 1	Model 2	Residual 2
6.7	4.5	7.3	-2.8	6.69	-2.19
9.0	11.0	7.3	3.7	8.41	2.59
7.5	6.0	7.3	-1.3	6.69	-0.69
5.0	8.0	7.3	0.7	7.51	0.49

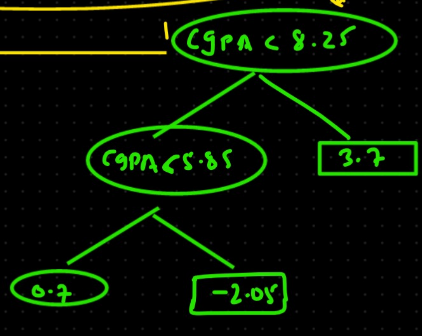
[CGPA, Res1]

SS → train

DT1

Final Pred ⇒ Base Model + eta * DT1

avg. LR first DT



Moto ⇒ Res ↓ ↓

Res ⟶ 0

Boosting

$\Rightarrow 7.3 + (0.3) * (-2.05) = 6.69$

$\Rightarrow 7.3 + 0.3 * (3.7) = 8.41$

$\Rightarrow 7.3 + 0.3 * (-2.05) = 6.69$

$\Rightarrow 7.3 + 0.3 * (0.7) = 7.51$

no_Base_estimator = 10, 15, 20 . .

Final result = BM + eta * DT$_1$ + eta * DT$_2$ - - + eta * DT$_n$

Gredient boost => ① XGB =>
② LGB =>
③ Catboost => } Library

Lin, non-Lin

ML => Data => various Less chance
Single
x

1980, 1970

Early => Linreg, logreg
naivebayes } => specific data

{SVM, RF, AB} => General Diff 1990-2000

overfitting, scalable Data
Performance => 2000 => Big data

XG Boost => Data => ML + SW

Early stage kaggle opensource

XGBoost
Performance speed flexibility HW + OS

① flexibility => ① cross platform
② Multiple language support
③ Integration with other library and tools
④ Support all kind of ML problems

Python
Java

feature

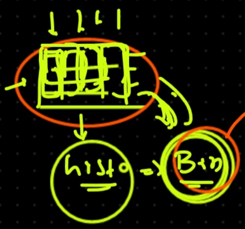
② Speed ⇒

① Parallel processing ⇒ CPU - GPU ⇒ Boosting ⇒

multiple cors

Process

② Optimized data structure

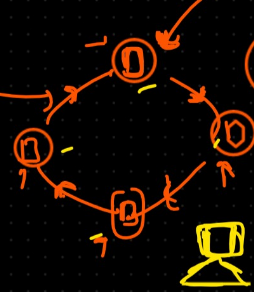


③ Cache awareness ⇒

④ Out of core computing → Data chunk → CPU

accessing ⇒ 1st, 2nd, 3rd, 4th, 5th

⑤ Distributed computing



big data

Master - Slave

⑥ GPU_support

(Graphical processing unit)

DL → {matrix multiplication}

Gaming → highend

Optimization ⇒ hw

system { App App App / OS / hw }

③ Performance

① Regularized Learning objective

Additive Learning fn

$$\Rightarrow \{ f_0 \Rightarrow B + \alpha M_1 + \alpha M_2 + \ldots \alpha M_3 \}$$

= { Regularization? → overfitting loss

classification

② Handling missing value ✗

③ Sparsity aware split finding

④ Efficient split finding {weighted quantile sketch + approximate tree learning}

⑤ tree pruning

post, pre

XGB Regression → Similarity score → GAIN

XGB classification

GB classification

web mus → next week

ML

Supervised

Regression
target => num
LR
SVR
DTR
RFR
XGR
GBR
KNNR
Distance
(lazy learner)

Classification
target => cat
log Reg
SVC
DTC
RFC
XGC
GBC
KNNC

Naive bayes
Prob => text classification
spam ham classification
{text preprocessing / text representation} NLP => ML

Unsupervised
clustering
group
- K mean, K mean++
- Hierarchical
- DBscan
wep

PCA => DF
lagrange → power
TSNE
LDA

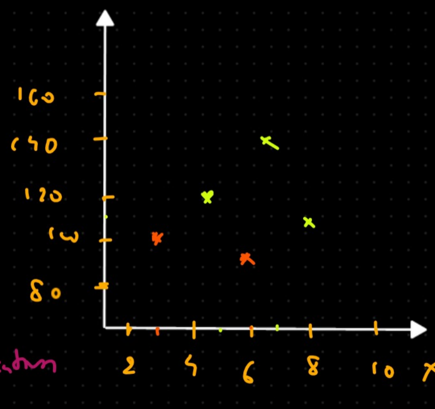
1500, 9520

Semi supervised
(custom)

K - Nearest -- Neighbor

Data Point

cgpa	iq	Placement
8	110	1
5	120	1
7	140	1
6	90	0
3	100	0
9	130	?



2_class

binary classification

K-NN

odd → 1, 3, 5, 7, 9, 11

even → 2, 10, 12, 6, 8, 4

$\boxed{k=3}$ Calculate the Distance

Query Point to other all the Points

$$\left\{ \begin{array}{l} (9-8)^2 = (1)^2 = 1 \\ (8-9)^2 = (-1)^2 = 1 \end{array} \right\}$$

① Eculidian Distance

② Manhattan Distance

$Q \to DT_1 = \sqrt{(8-9)^2+(110-130)^2} = \sqrt{1+400} = \sqrt{401} = 20.02$

$Q \to DT_2 = \sqrt{(9-5)^2+(120-130)^2} = \sqrt{16+100} = \sqrt{116} = 10.77$

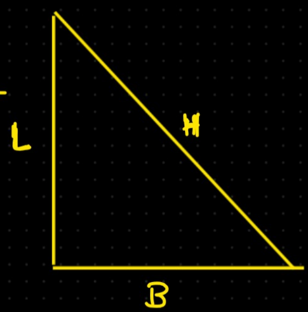
$Q \to DT_3 = \sqrt{(9-7)^2+(140-130)^2} = \sqrt{4+100} = \sqrt{104} = 10.19$

$Q \to DT_4 = \sqrt{(9-6)^2+(90-130)^2} = \sqrt{9+1600} = \sqrt{1609} = 40.11$

$Q \to DT_5 = \sqrt{(9-3)^2+(100-130)^2} = \sqrt{36+900}$ 20 30.59

$= \sqrt{536}$

k=0



$H^2 = L^2 + B^2$

$\boxed{H = \sqrt{L^2 + B^2}}$

$\boxed{H = \sqrt{(y_2-y_1)^2 + (x_2-x_1)^2}}$

Sort

0 → 0

1 → 3

1 ←	Q point ⟺ DT_1 =	20.02	10.19
1 ←	Q Point ⟺ DT_2 =	10.77	10.77
1 ←	Q Point ⟺ DT_3 =	10.19	20.02
	Q Point ⟺ DT_4 =	40.11	30.59
	Q Point ⟺ DT_5 =	30.59	40.4

majority count

$(y_1-y_2)^2$

(y_2-y_1)

⇒ k=3

~~Learning~~ Distance

Non-parametric

Lazy Learner

Regression ⇒ Average

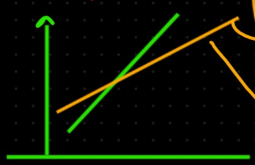
Data

train

test

Linear reg

$y = mx + c$ ⇒ gredient Decent (m,c)



1 ⇒ { 0 → 0.5, 1 → 0.5 }

2 → { 0 → 0.7, 1 → 0.3 }

Mode

3 → { 0 → 0.8, 1 → 0.3 }

{ Hard voting }

{ Soft voting } → Prob.

KNN

i am not doing learning

(Non-Param)

Model ⇒

Distance

M/A

Frac/o/p

row

3 NN

5 NN

Sorting Distance

K (Nearest Neighbor)

3

5

$f_1 \quad f_2 \quad f_3 \quad f_4 \quad f_5 \quad Y$

$x_1' \quad x_2' \quad x_3' \quad x_4' \quad x_5'$

$$\sqrt{(x_1 - x_1')^2 + (x_2 - x_2')^2 + (x_3 - x_3')^2 + (x_4 - x_4')^2 + (x_5 - x_5')^2}$$

Optimize

{K-D tree, ball tree}

{arrangement}

=> Next class